

(19)



(11)

EP 3 602 369 B1

(12)

EUROPEAN PATENT SPECIFICATION

(45) Date of publication and mention of the grant of the patent:
30.03.2022 Bulletin 2022/13

(21) Application number: **18711988.8**

(22) Date of filing: **26.03.2018**

(51) International Patent Classification (IPC):
G06F 21/55^(2013.01)

(52) Cooperative Patent Classification (CPC):
G06F 21/55

(86) International application number:
PCT/EP2018/057685

(87) International publication number:
WO 2018/178034 (04.10.2018 Gazette 2018/40)

(54) **ANOMALY DETECTION FOR COMPUTER SYSTEMS**

ANOMALIEDETEKTION FÜR COMPUTERSYSTEME

DÉTECTION D'ANOMALIE POUR SYSTÈMES INFORMATIQUES

(84) Designated Contracting States:
**AL AT BE BG CH CY CZ DE DK EE ES FI FR GB
GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO
PL PT RO RS SE SI SK SM TR**

(30) Priority: **30.03.2017 EP 17164005**

(43) Date of publication of application:
05.02.2020 Bulletin 2020/06

(73) Proprietor: **British Telecommunications public
limited company
London E1 8EE (GB)**

(72) Inventors:
• **DANIEL, Joshua
London EC4V 5BT (GB)**
• **DU, Xiaofeng
London EC4V 5BT (GB)**

(74) Representative: **British Telecommunications
public limited company
Intellectual Property Department
9th Floor
One Braham
Braham Street
London E1 8EE (GB)**

(56) References cited:
**US-A1- 2012 304 007 US-A1- 2014 067 734
US-A1- 2015 227 741 US-A1- 2016 350 173**

- **WU JIA ET AL: "Hierarchical Temporal Memory
Method for Time-Series-Based Anomaly
Detection", 2016 IEEE 16TH INTERNATIONAL
CONFERENCE ON DATA MINING WORKSHOPS
(ICDMW), IEEE, 12 December 2016 (2016-12-12),
pages 1167-1172, XP033055893, DOI:
10.1109/ICDMW.2016.0168**

Note: Within nine months of the publication of the mention of the grant of the European patent in the European Patent Bulletin, any person may give notice to the European Patent Office of opposition to that patent, in accordance with the Implementing Regulations. Notice of opposition shall not be deemed to have been filed until the opposition fee has been paid. (Art. 99(1) European Patent Convention).

EP 3 602 369 B1

Description

[0001] The present invention relates to the detection of anomalous operation of a computer system. In particular, it relates to the detection of non-compliance based on anomalies.

[0002] Computer systems are susceptible to misuse, hijack or malicious software and/or access that can lead to harm including: data loss; the execution or perpetuation of malicious software; data theft; misappropriation of information or computing resources; interruption, denial or degradation of service; or other harms as will be familiar to those skilled in the art.

[0003] Techniques to detect and remediate inappropriate, unauthorised or undesirable use of a computer system are focussed on recognising such use and reacting accordingly. Such techniques therefore require constant monitoring of a computer system in use to spot problematic operation or use based on rules. While this approach may be effective, it is very resource intensive and cumbersome to apply to an entire computer system or a set of disparate computer systems. For example, where two disparate systems are required to comply with a single set of rules it can be necessary to implement multiple different monitoring mechanisms, one of each system, to accommodate technical differences therebetween.

[0004] It would be advantageous to monitor the compliance of computer systems with compliance rules without the aforementioned disadvantages.

[0005] US patent publication US 2014/067734 A1 describes detecting patterns and sequences associated with an anomaly in predictions made a predictive system.

[0006] US patent publication US 2015/227741 A1 describes techniques purported to enable computers to efficiently determine if they should run a program based on an immediate (i.e., real-time, etc.) analysis of the program.

[0007] US patent publication US 2012/304007 A1 describes methods and apparatus purported for use in identifying abnormal behavior in a control system

[0008] US patent publication US 2016/350173 A1 describes an anomaly detection and resolution system purportedly for automatically detecting and resolving anomalies in computing environments

[0009] The paper "Hierarchical Temporal Memory for Time-Series-Based Anomaly Detection" (Wu Jia et al., 2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW), IEEE, 12 December 2016, pages 1167-1172, XP033055893) describes an experiment for detecting anomalies in time series data.

[0010] The present invention accordingly provides, a computer implemented method to detect a computer system in execution operating in a manner that is not compliant with a definition of a set of compliant operations, the method comprising: receiving a first set of records for the computer system, each record detailing an occurrence in the computer system during a first predetermined time period; generating a sparse distributed representation of the first set of records to form a training set for a hierarchical temporal memory (HTM); training the HTM based on the training set in order that the trained HTM provides a model of the operation of the computer system during the predetermined time period; selecting at least a subset of operations in the set of compliant operations and causing the invocation of each operation of the subset in the computer system over a second predetermined time period to generate a second set of records of occurrences in the computer system; generating a sparse distributed representation of the second set of records to form an input set for the trained HTM; executing the trained HTM based on the input set to determine a degree of recognition of the records of the input set; and responsive to a determination that a degree of recognition of one or more records of the input set is below a threshold degree, identifying the operation of the computer system as non-compliant.

[0011] Preferably the method further comprises: in response to an identification that the operation of the computer system is non-compliant, implementing a protective measure to protect against a malicious operation of the computer system.

[0012] Preferably the protective measure includes one or more of: causing a cessation of operation of the computer system; generating an event indicating the non-compliance of the computer system; suspending operation of the computer system; executing one or more protective and/or remedial software components in the computer system.

[0013] The present invention accordingly provides, in a second aspect, a computer system including a processor and memory storing computer program code for performing the steps of the method set out above.

[0014] The present invention accordingly provides, in a third aspect, a computer program element comprising computer program code to, when loaded into a computer system and executed thereon, cause the computer to perform the steps of the method set out above.

[0015] Embodiments of the present invention will now be described, by way of example only, with reference to the accompanying drawings, in which:

Figure 1 is a block diagram of a computer system suitable for the operation of embodiments of the present invention;

Figure 2 is a component diagram of an exemplary arrangement of a system in accordance with embodiments of the present invention;

Figure 3 is a component diagram of an exemplary arrangement of a compliance engine configured to train a hierarchical temporal memory in accordance with embodiments of the present invention;

Figure 4 is a component diagram of an exemplary arrangement of a compliance engine configured to

determine compliance of a computer system with operations in a compliant operation set in accordance with embodiments of the present invention; and

Figure 5 is a flowchart of a method to detect a computer system in execution operating in a manner that is not compliant with a definition of a set of compliant operations in accordance with embodiments of the present invention.

[0016] Figure 1 is a block diagram of a computer system suitable for the operation of embodiments of the present invention. A central processor unit (CPU) 102 is communicatively connected to a storage 104 and an input/output (I/O) interface 106 via a data bus 108. The storage 104 can be any read/write storage device such as a random access memory (RAM) or a non-volatile storage device. An example of a non-volatile storage device includes a disk or tape storage device. The I/O interface 106 is an interface to devices for the input or output of data, or for both input and output of data. Examples of I/O devices connectable to I/O interface 106 include a keyboard, a mouse, a display (such as a monitor) and a network connection.

[0017] Figure 2 is a component diagram of an exemplary arrangement of a system in accordance with embodiments of the present invention. A computer system 200 is provided as a physical, virtual or combination computer system for which a state of operational compliance is to be determined. A determination of a state of operational compliance for the computer system 200 includes determining if the computer system 200 is compliant with rules of operation defined for the computer system. Such rules can include, inter alia: a definition of software, services, protocols and/or functions that can be used and/or executed by the computer system 200; a definition of a volume, quantity, extent or proportion of one or more resources that can be reserved, employed, accessed or otherwise utilised by the computer system 200; temporal restrictions on operation of all or part of the computer system 200; user access controls to all or part of the computer system 200; network communication that is permitted and/or precluded including definitions of rules relating to network endpoints, protocols, services, appliances and the like that may or may not be employed by or used with the computer system; a definition of data that can or cannot be stored in, with or by the computer system; a definition of one or more technical standards that must be employed or must not be employed by the computer system including standard protocols, encryption, access control mechanisms, software and/or data verification, resource consumption, standard security facilities including network security, antivirus, intrusion detection and the like, and other such standards as will be apparent to those skilled in the art; and other compliance requirements.

[0018] It will be appreciated that the computer system 200 can be a complete computer system such as illus-

trated in Figure 1 or can be a part of a computer system 200 such as a software component or application executing on or with a computer system, a service provided at least in part by a computer system, or a network of multiple computer systems in communication.

[0019] Figure 2 further includes a compliance engine 202 arranged to access the computer system 200 to determine a state of operational compliance of the computer system 200. The compliance engine 202 thus generates a compliance determination 204 for the computer system 200. Notably, the compliance determination 204 will relate to operation of the computer system 200 for a defined period of time (since a compliant computer system could become subsequently non-compliant). Accordingly, in some embodiments the compliance engine 202 operates on a continuous basis to determine a state of operational compliance for the computer system 200.

[0020] Figure 3 is a component diagram of an exemplary arrangement of a compliance engine 202 configured to train a hierarchical temporal memory (HTM) 320 in accordance with embodiments of the present invention. The compliance engine 202 receives a first log 310 from the computer system 200. The first log 310 is a set of records for the computer system 200 where each record details an occurrence in the computer system during a defined time period while the computer system 200 is in operational use. Operational use refers to a normal use of the computer system 200 by users, systems and the like. For example, the log 310 can include records relating to: user access; data storage; network communication; systems configuration; software execution; errors; warnings; configuration changes; software events; and other records of occurrences in the computer system 200 as will be apparent to those skilled in the art.

[0021] The compliance engine 202 uses the records in the first log 310 to constitute training data inputs for training a HTM 320. The HTM 320 is a machine learning construct based on principles first described by Jeff Hawkins in "On Intelligence" (2004, Times Books, ISBN 0-8050-7456-2) and described in detail by Numenta in "Hierarchical Temporal Memory including HTM Cortical Learning Algorithms" (Numenta, 12 September 2011). The principles of, implementation of and operation of HTM 320 are beyond the scope of this description and are nonetheless available to the skilled person through existing publications including the papers and books below, each and/or all of which are usable by a skilled person to implement the HTM 320 and other associated features for embodiments of the present invention:

- "Biological and Machine Intelligence (BAMI) - A living book that documents Hierarchical Temporal Memory (HTM)" (Numenta, Inc., March 8, 2017) (retrieved March 29, 2017) available from numenta.com/assets/pdf/biological-and-machine-intelligence/BAMI-Complete.pdf
- "Porting HTM Models to the Heidelberg Neuromorphic Computing Platform" (Billaudelle, S. & Ahmad,

- S., May 2015) available from Cornell University Library (citation arXiv:1505.02142) and arxiv.org/abs/1505.02142
- "Encoding Data for HTM Systems" (Purdy, S., February 2016) available from Cornell University Library (citation arXiv:1602.05925) and arxiv.org/abs/1602.05925
 - "Properties of Sparse Distributed Representations and their Application To Hierarchical Temporal Memory" (Ahmad, S. & Hawkins, J., March 2015) available from Cornell University Library (citation arXiv:1503.07469) and arxiv.org/abs/1503.07469
 - "How Do Neurons Operate on Sparse Distributed Representations? A Mathematical Theory of Sparsity, Neurons and Active Dendrites" (Ahmad, S. & Hawkins, J., January 2016) available from Cornell University Library (citation arXiv:1601.00720) and arxiv.org/abs/1601.00720
 - "Real-Time Anomaly Detection for Streaming Analytics" (Ahmad, S. & Purdy, S., July 2016) available from Cornell University Library (citation arXiv:1607.02480) and arxiv.org/abs/1607.02480
 - "Evaluating Real-time Anomaly Detection Algorithms - the Numenta Anomaly Benchmark" (Lavin, A. & Ahmad, S., October 2015) available from Cornell University Library (citation arXiv:1510.03336) and arxiv.org/abs/1510.03336
 - "The HTM Spatial Pooler: A Neocortical Algorithm for Online Sparse Distributed Coding" (Cui, Y., Ahmad, S. & Hawkins, J., February 2017) available from Cold Spring Harbor Laboratory bioRxiv (citation doi.org/10.1101/085035) and biorxiv.org/content/biorxiv/early/2017/02/16/085035.full.pdf
 - "Continuous Online Sequence Learning with an Unsupervised Neural Network Model" (Cui, Y., Ahmad, S. & Hawkins, K., November 2016) published in *Published in Neural Computation* (November 2016, Vol 28. No. 11) and available from numenta.com/papers/continuous-online-sequence-learning-with-an-unsupervised-neural-network-model/
 - "Why Neurons Have Thousands of Synapses, A Theory of Sequence Memory in Neocortex" (Hawkins, J. & Ahmad, S., March 2016) published in *Frontiers in Neural Circuits* (10 (2016) 1-13, doi:10.3389/fncir.2016.00023) and available from numenta.com/papers/why-neurons-have-thousands-of-synapses-theory-of-sequence-memory-in-neocortex/

[0022] At a very high level, in one embodiment, the HTM 320 is implementable logically as a hierarchy of functional nodes. The hierarchy of nodes in the HTM 320 are suitable for identifying coincidences in a temporal sequence of input patterns received at an input layer in the hierarchy, with interconnections between the layers permitting such identifications to take place also at each other level in the hierarchy. In addition to an identification of coincidences by nodes in the HTM 320, temporal re-

lationships between coincidences can also be identified. Thus, in a purely exemplary arrangement, a first set of similar patterns occurring before a second set of similar patterns can be resolved to a coincidence (of the first set) with a temporal relationship to a coincidence (of the second set). The coincidences and temporal relations learned at each of many levels in the hierarchical HTM 320 provide for subsequent recognition, by the HTM 320, of a conforming temporal sequence of input patterns and non-conformant sequences. Thus, the HTM 320 can be said to operate in: a learning mode of operation in which coincidences and relationships between coincidences are learned by adaptation of the HTM 320; and an inference mode of operation in which the HTM 320 is executed (by which it is meant that the HTM 320 is applied) to process one or more inputs to determine a degree of recognition of the inputs by the HTM 320 based on what has been learned by the HTM 320. Recognition can be based on a determination, by nodes in the HTM 320, of a set of probabilities that an input belongs to one or more known or recognised coincidences in the trained HTM 320, and probabilities that inputs represent a recognised temporal group of coincidences.

[0023] When applied in embodiments of the present invention, the HTM 320 has two key features: firstly, the HTM 320 is trained based on the first log 310 to represent a model of the operation of the computer system 200 in operational use; and secondly the HTM 320 can determine whether subsequent data sets are recognisable to the HTM 320 and thus bear similarity to the operation of the computer system 200 in operational use.

[0024] While the HTM 320 has been described, by way of overview, structurally here, it will be appreciated that its implementation can be a logical representation or approximation of such a structure including a mathematical implementation employing, for example, linear algebra and/or parallel processing means for implementation.

[0025] The HTM 320 is trained by a HTM trainer 314 which is a hardware, software, firmware or combination component adapted to undertake the training of the HTM 320. It will be appreciated, on the basis of the above referenced papers and books, that the HTM 320 preferably operates on the basis of a sparse distributed representation (SDR) 312 of data. For example, an SDR can be a binary representation of data comprised of multiple bits in which only a small percentage of the bits are active (i.e. binary 1). The bits in these representations have semantic meaning and meanings are distributed across the bits. SDR is described in "Sparse Distributed Representations" (Numenta, available from github.com/numenta/nupic/wiki/Sparse-Distributed-Representations and accessed on 29th March 2017). Further, the principles underlying SDR are also described in "Sparse coding with an overcomplete basis set: A strategy employed by V1?" (Olshausen, B.A., Field, D.J., 1997, *Vision Research*, 37:3311-3325). Accordingly, the records in the first log 310 are initially encoded to a SDR by a suitable encoder. Notably, the encoder is configured to set bits in

a SDR 312 for a record based on a semantic meaning of the bits and thus the encoder is specifically configured to encode each record in to a SDR 312 based on semantic meaning of some aspect of the record including, for example, one or more of: a content of the record; characteristics of the record such as its length, origin, when it was received, how it was created, what created it etc.; what the record means, what it indicates, what consequence may ensue as a result of an occurrence recorded by the record etc.; and other aspects as will be apparent to those skilled in the art.

[0026] Thus, in use, the compliance engine 202 trains the HTM 320 using SDR representation 312 of records received in the first log 310 for the computer system 200 in operational use. Accordingly, following training, the HTM 320 can be said to constitute a model or record of the operation of the computer system 200 during the time period for which the first log 310 was received. This model is subsequently used to detect an anomalous operation of the computer system 200 vis a vis a set of compliant operations as will be described with respect to Figure 4.

[0027] Figure 4 is a component diagram of an exemplary arrangement of a compliance engine 202 configured to determine compliance of a computer system 200 with operations in a compliant operation set 444 in accordance with embodiments of the present invention. The compliant operation set 444 is a set of operations that can be invoked, caused to be invoked or effected, on the computer system 200. For example, the operations can be: user operations such as a user access to a system, resource, component or the like of computer system 200; a configuration operation such as a change to a configuration of a software, hardware, firmware, access control, network, storage, processing, application or other resource of the computer system 200; a network operation such as a sending or receiving of network communication; a software, service, function or resource installation, activation, deinstallation or deactivation; a change to a volume, quantity or amount of one or more resources; a storage, retrieval or transformation of information in or by the computer system 200; or other operations as will be apparent to those skilled in the art. Notably, the operations in the compliant operation set 444 are operations that specifically comply with any compliance requirements for the computer system 200. In a preferred embodiment, the compliant operation set 444 includes operations covering, in combination, substantially all aspects of the computer system 200 for which compliance requirements are defined. At least some subset of the operations in the compliant operation set 444 are selected by an operation selector 442. Such a selection can be random, defined by a rule, schedule or may involve the selection of operations from each of a plurality of aspects of operation of the computer system 200. Once selected, an operation invoker 440 triggers, causes or effects an invocation of each operation by the computer system 200. Such invocation can be effected by the invoker 440 by way of an administrative interface to the

computer system 200 such as a privileged access arrangement by the invoker 440. Thus, the invoker 440 causes operations from the compliant operation set 444 to be invoked by the computer system 200.

[0028] The invocation of the compliant operations by the computer system 200 will generate records as a second log 410 corresponding to log, trace, event or other information relating to the operation of the computer system 200 when undertaking the operations invoked by the invoker 440. Thus, the second log 410 reflects the operation of the computer system 200 undertaking compliant operations. Notably, the invocation and undertaking of compliant operations by the computer system 200 invoked by the invoker 440 can occur at the same time as the computer system 200 continues with its normal operation such that records reflecting the operational use of the computer system 200 may be included in the second log 410 along with records arising from the compliant operations.

[0029] Subsequently, an SDR 412 of the records of the second log is generated by an encoder substantially as previously described with respect to the first log 310. A HTM executer 414 then executes the HTM 320 (now trained by way of the arrangement of Figure 3) in an inference or recognition mode of operation. In this mode of operation, the HTM 320 determines a degree of recognition of each SDR data item input to it based on its learned model of the operational use of the computer system 200. Based on this degree of recognition the HTM 320 also identifies anomalies as SDR inputs that are not recognised by the trained HTM 320.

[0030] The HTM 320, modelling the computer system 200 in operational use, will indicate a strong degree of recognition of SDR for records of the second log 410 arising from compliant operations invoked in the computer system 200 if the computer system 200 in operational use is compliant. If, however, the computer system 200 in operational use is non-compliant then the model of the computer system 200 constituted by the trained HTM 320 is a model of a non-compliant system. Accordingly, in such circumstances, the SDR of records from the second log 410 arising from the invocation of *compliant* operations will not be recognised, or will be less significantly recognised, by the HTM 320. Indeed such compliant operations will be identified by the HTM 320 as anomalies because they appear anomalous to the learned operational use of the computer system 200 which is non-compliant. That is to say that a degree of recognition of compliant operations by the HTM 320 trained based on a non-compliant computer system 200 will be lower than a degree of recognition where the HTM 320 is trained based on a compliant computer system 200. Thus, according to the arrangement in embodiments of the present invention, execution of the HTM 320 to recognise (or not) SDR 412 from the second log 410 serves to identify if the computer system 200 in operational use is compliant. Accordingly, a compliance determination 204 can be made by the compliance engine.

[0031] Figure 5 is a flowchart of a method to detect a computer system 200 in execution operating in a manner that is not compliant with a definition of a set of compliant operations 444 in accordance with embodiments of the present invention. Initially, at step 502, the compliance engine 202 receives a first log 310 for the operation of the computer system 200 over a predetermined time period in an operational use. At step 504 a SDR 314 of the records in the first log 310 is generated, such as by an encoder. At step 506 the HTM 320 is trained based on the SDR 314. At step 508, compliant operations are selected by the selector 442 from a compliant operation set 444. At step 510 the operation invoker 440 invokes the selected compliant operations in the computer system 200 to generate a second log 410 of records. At step 512 the compliance engine 202 receives the second log 410 and encodes the records of the second log 410 into a SDR 412 at step 514. At step 516 the HTM 320 is executed with the SDR 412 of the second log 410 and non-compliance is determined at step 518 based on a degree of recognition of the HTM 320 of the SDR 412 of the second log.

[0032] Where the computer system 200 is determined to operate in non-compliance with requirements, responsive action can be taken. For example, such action can include protective measures to protect against a malicious operation of the computer system. In some embodiments, protective measures can include, for example: causing a cessation of operation of the computer system 200; generating an event indicating the non-compliance of the computer system 200; suspending operation of the computer system 200; executing one or more protective and/or remedial software components in the computer system. For example: a malware scanner could be invoked; a review of user access control and logs can be undertaken; a firewall can be installed or reconfigured; antivirus software can be invoked or reconfigured; and other such operations as will be apparent to those skilled in the art.

[0033] Insofar as embodiments of the invention described are implementable, at least in part, using a software-controlled programmable processing device, such as a microprocessor, digital signal processor or other processing device, data processing apparatus or system, it will be appreciated that a computer program for configuring a programmable device, apparatus or system to implement the foregoing described methods is envisaged as an aspect of the present invention. The computer program may be embodied as source code or undergo compilation for implementation on a processing device, apparatus or system or may be embodied as object code, for example.

[0034] Suitably, the computer program is stored on a carrier medium in machine or device readable form, for example in solid-state memory, magnetic memory such as disk or tape, optically or magneto-optically readable memory such as compact disk or digital versatile disk etc., and the processing device utilises the program or a

part thereof to configure it for operation. The computer program may be supplied from a remote source embodied in a communications medium such as an electronic signal, radio frequency carrier wave or optical carrier wave. Such carrier media are also envisaged as aspects of the present invention.

[0035] It will be understood by those skilled in the art that, although the present invention has been described in relation to the above described example embodiments, the invention is not limited thereto and that there are many possible variations and modifications which fall within the scope of the invention.

[0036] The scope of the present invention includes any novel features or combination of features disclosed herein. The applicant hereby gives notice that new claims may be formulated to such features or combination of features during prosecution of this application or of any such further applications derived therefrom. In particular, with reference to the appended claims, features from dependent claims may be combined with those of the independent claims and features from respective independent claims may be combined in any appropriate manner and not merely in the specific combinations enumerated in the claims.

Claims

1. A computer implemented method to detect a computer system in execution operating in a manner that is not compliant with a definition of a set of compliant operations (444), the method comprising:

receiving (502) a first set of records (310) for the computer system, each record detailing an occurrence in the computer system during a first predetermined time period;
 generating (504) a sparse distributed representation (312) of the first set of records to form a training set for a hierarchical temporal memory, HTM, (320);
 training (506) the HTM based on the training set in order that the trained HTM provides a model of the operation of the computer system during the predetermined time period;
 selecting (508) at least a subset of compliant operations in the set of compliant operations;
 invoking (510) each operation of the selected subset of compliant operations in the computer system over a second predetermined time period to generate a second set of records (410) of occurrences in the computer system;
 generating (514) a sparse distributed representation (412) of the second set of records to form an input set for the trained HTM;
 executing (516) the trained HTM based on the input set to determine a degree of recognition of the records of the input set; and

responsive to a determination that a degree of recognition of one or more records of the input set is below a threshold degree, identifying (518) the operation of the computer system as non-compliant.

2. The method of claim 1 further comprising: in response to an identification that the operation of the computer system is non-compliant, implementing a protective measure to protect against a malicious operation of the computer system.

3. The method of claim 2 wherein the protective measure includes one or more of:

causing a cessation of operation of the computer system; generating an event indicating the non-compliance of the computer system; suspending operation of the computer system; executing one or more protective and/or remedial software components in the computer system.

4. A computer system including a processor and memory storing computer program code for performing the steps of any preceding claim.

5. A computer program element comprising computer program code to, when loaded into a computer system and executed thereon, cause the computer to perform the steps of a method as claimed in any of claims 1 to 3.

Patentansprüche

1. Computerimplementiertes Verfahren zum Detektieren, dass ein Computersystem bei Ausführung auf eine Weise arbeitet, die nicht mit einer Definition einer Menge von konformen Operationen (444) konform ist, wobei das Verfahren Folgendes umfasst:

Empfangen (502) einer ersten Menge von Datensätzen (310) für das Computersystem, wobei jeder Datensatz ein Auftreten in dem Computersystem während eines ersten vorbestimmten Zeitraums detailliert;

Erzeugen (504) einer spärlichen verteilten Repräsentation (312) der ersten Menge von Datensätzen zum Bilden einer Trainingsmenge für einen hierarchischen temporären Speicher, HTM, (320);

Trainieren (506) des HTM auf der Grundlage der Trainingsmenge, sodass der trainierte HTM ein Modell des Betriebs des Computersystems während der vorbestimmten Zeitspanne liefert; Auswählen (508) mindestens einer Untermenge von konformen Operationen in der Menge von

konformen Operationen;

Aufrufen (510) von jeder Operation der ausgewählten Untermenge von konformen Operationen in dem Computersystem über eine zweite vorbestimmte Zeitspanne, um eine zweite Menge von Datensätzen (410) von Auftretensfällen in dem Computersystem zu erzeugen;

Erzeugen (514) einer spärlichen verteilten Repräsentation (412) der zweiten Menge von Datensätzen zum Bilden einer Eingabemenge für den trainierten HTM;

Ausführen (516) des trainierten HTM auf der Grundlage der Eingabemenge zum Bestimmen eines Erkennungsgrads der Datensätze der Eingabemenge; und

als Reaktion auf eine Bestimmung, dass ein Erkennungsgrad von einem oder mehreren Datensätzen der Eingabemenge unter einem Schwellengrad liegt, Identifizieren (518) des Betriebs des Computersystems als nicht-konform.

2. Verfahren nach Anspruch 1, ferner umfassend: als Reaktion auf eine Identifikation, dass der Betrieb des Computersystems nicht-konform ist, Implementieren einer Schutzmaßnahme zum Schützen gegenüber einem schädlichen Betrieb des Computersystems.

3. Verfahren nach Anspruch 2, wobei die Schutzmaßnahme eine oder mehrere der folgenden beinhaltet: Bewirken einer Beendigung des Betriebs des Computersystems; Erzeugen eines Ereignisses, das die Nichtkonformität des Computersystems angibt; Suspendieren des Betriebs des Computersystems; Ausführen einer oder mehrerer Schutz- und/oder Behebungs-Softwarekomponenten in dem Computersystem.

4. Computersystem, das einen Prozessor und Speicher zum Speichern von Computerprogrammcode zum Durchführen der Schritte nach einem vorhergehenden Anspruch beinhaltet.

5. Computerprogrammelement, das Computerprogrammcode umfasst zum, wenn in ein Computersystem geladen und darauf ausgeführt, Veranlassen des Computers, die Schritte eines Verfahrens nach einem der Ansprüche 1 bis 3 durchzuführen.

Revendications

1. Procédé mis en œuvre par ordinateur pour détecter un système informatique en cours d'exécution fonctionnant d'une manière non conforme à une définition d'un ensemble d'opérations conformes (444), le procédé comprenant :

- la réception (502) d'un premier ensemble d'enregistrements (310) pour le système informatique, chaque enregistrement détaillant une occurrence dans le système informatique durant une première période de temps prédéterminée ; la génération (504) d'une représentation distribuée clairsemée (312) du premier ensemble d'enregistrements pour former un ensemble d'apprentissage pour une mémoire temporelle hiérarchique, HTM, (320) ; l'apprentissage (506) de la HTM en fonction de l'ensemble d'apprentissage afin que la HTM formée fournisse un modèle du fonctionnement du système informatique durant la période de temps prédéterminée ; la sélection (508) d'au moins un sous-ensemble d'opérations conformes dans l'ensemble d'opérations conformes ; l'appel (510) de chaque opération du sous-ensemble sélectionné d'opérations conformes dans le système informatique durant une seconde période de temps prédéterminée pour générer un second ensemble d'enregistrements (410) d'occurrences dans le système informatique ; la génération (514) d'une représentation distribuée clairsemée (412) du second ensemble d'enregistrements pour former un ensemble d'entrée pour la HTM formée ; l'exécution (516) de la HTM formée en fonction de l'ensemble d'entrée pour déterminer un degré de reconnaissance des enregistrements de l'ensemble d'entrée ; et en réponse à une détermination qu'un degré de reconnaissance d'un ou de plusieurs enregistrements de l'ensemble d'entrées est inférieur à un degré seuil, l'identification (518) d'un fonctionnement non conforme du système informatique.
2. Procédé selon la revendication 1 comprenant en outre : en réponse à une identification d'un fonctionnement non conforme du système informatique, la mise en œuvre d'une mesure de protection contre un fonctionnement malveillant du système informatique.
3. Procédé selon la revendication 2 dans lequel la mesure de protection comporte un ou plusieurs des mesures suivantes : l'arrêt du fonctionnement du système informatique ; la génération d'un événement indiquant la non-conformité du système informatique ; la mise en suspens du fonctionnement du système informatique ; l'exécution d'un ou plusieurs composants logiciels de protection et/ou de réparation dans le système informatique.
4. Système informatique comprenant un processeur et

une mémoire stockant un code de programme informatique pour réaliser les étapes selon n'importe quelle revendication précédente.

5. Elément de programme d'ordinateur comprenant un code de programme d'ordinateur pour, lorsqu'il est chargé dans un système informatique et exécuté sur celui-ci, amener l'ordinateur à réaliser les étapes d'un procédé selon l'une quelconque des revendications 1 à 3.

FIGURE 1

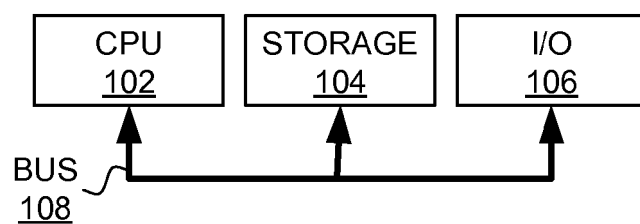


FIGURE 2

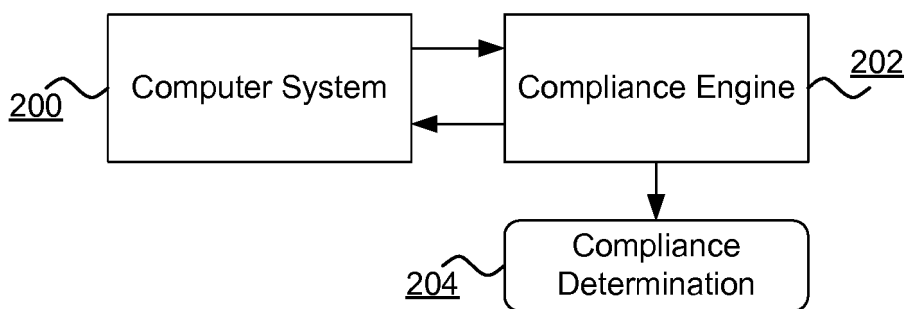


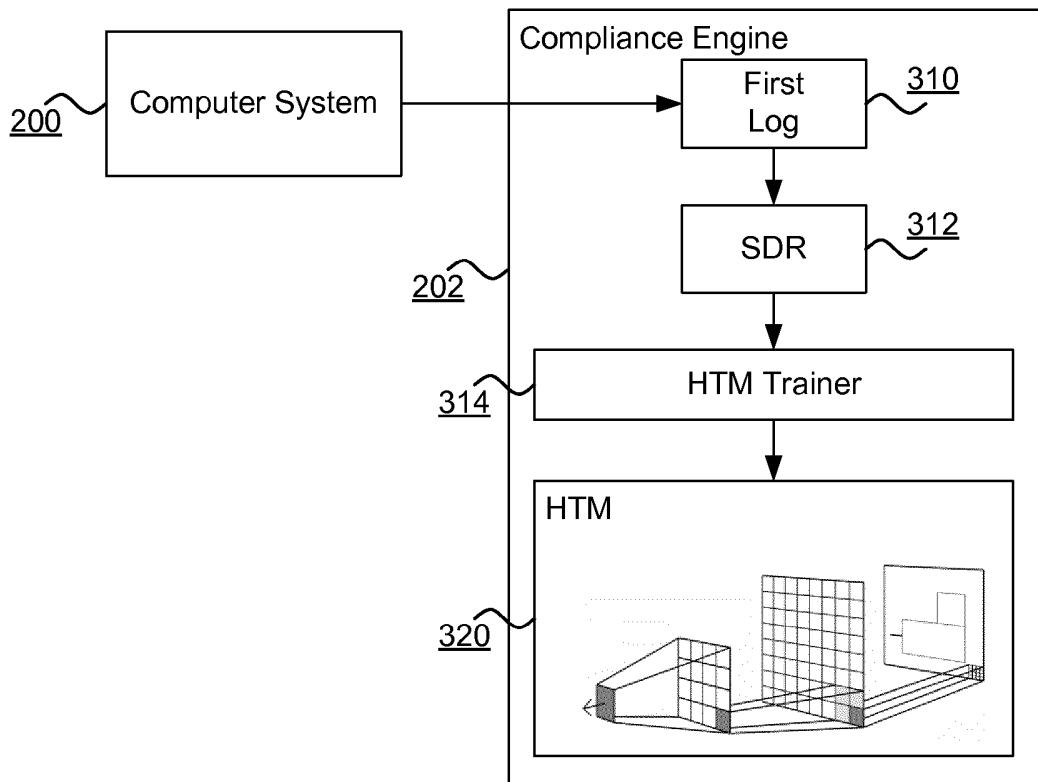
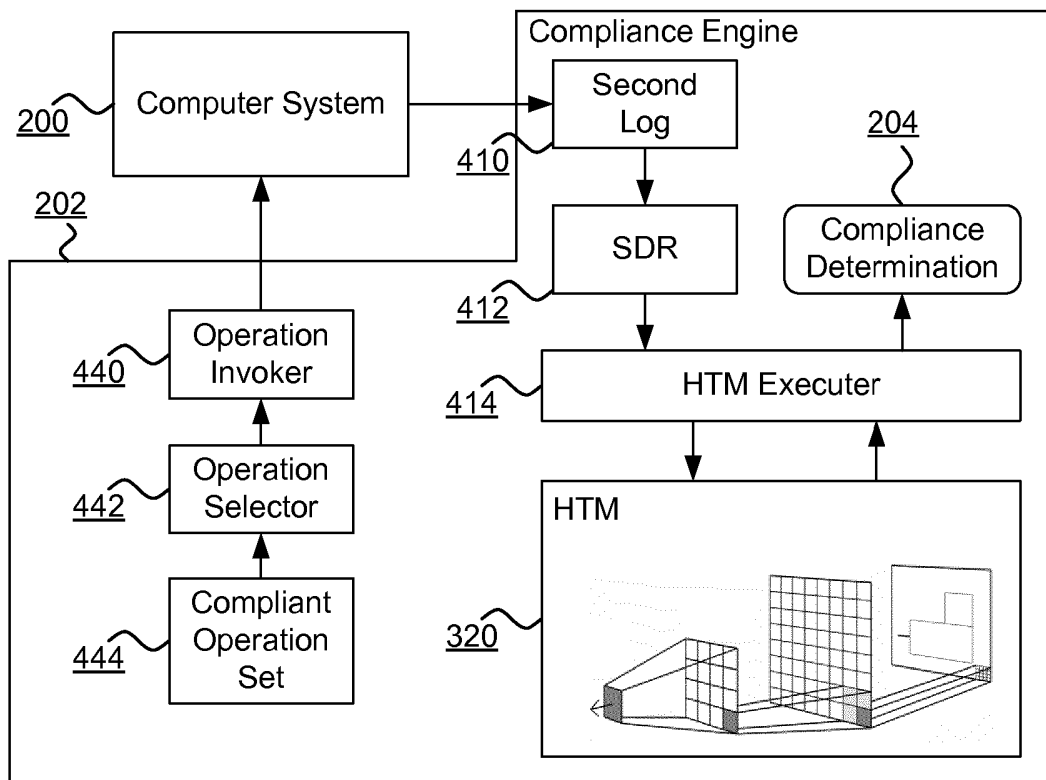
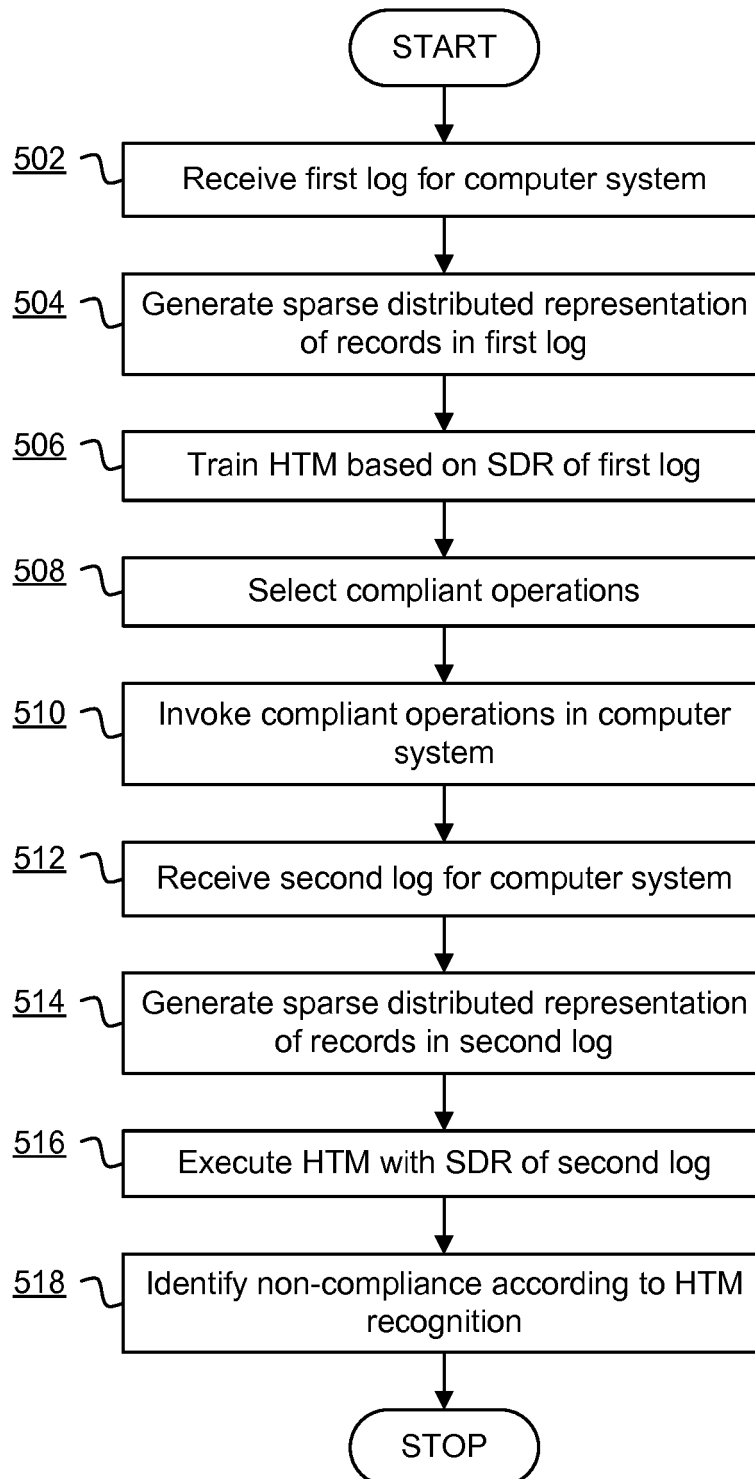
FIGURE 3**FIGURE 4**

FIGURE 5

REFERENCES CITED IN THE DESCRIPTION

This list of references cited by the applicant is for the reader's convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.

Patent documents cited in the description

- US 2014067734 A1 [0005]
- US 2015227741 A1 [0006]
- US 2012304007 A1 [0007]
- US 2016350173 A1 [0008]

Non-patent literature cited in the description

- Hierarchical Temporal Memory for Time-Series-Based Anomaly Detection. **WU JIA et al.** 2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW). IEEE, 12 December 2016, 1167-1172 [0009]
- **JEFF HAWKINS.** On Intelligence. Times Books, 2004 [0021]
- Hierarchical Temporal Memory including HTM Cortical Learning Algorithms. Numenta, 12 September 2011 [0021]
- Biological and Machine Intelligence (BAMI) - A living book that documents Hierarchical Temporal Memory (HTM). Numenta, Inc, 08 March 2017 [0021]
- Porting HTM Models to the Heidelberg Neuromorphic Computing Platform. **BILLAUELLE, S. ; AHMAD, S.** arXiv:1505.02142. Cornell University Library, May 2015 [0021]
- Encoding Data for HTM Systems. **PURDY, S.** arXiv:1602.05925. Cornell University Library, February 2016 [0021]
- Properties of Sparse Distributed Representations and their Application To Hierarchical Temporal Memory. **AHMAD, S. ; HAWKINS, J.** arXiv:1503.07469. Cornell University Library, March 2015 [0021]
- How Do Neurons Operate on Sparse Distributed Representations? A Mathematical Theory of Sparsity, Neurons and Active Dendrites. **AHMAD, S. ; HAWKINS, J.** arXiv:1601.00720. Cornell University Library, January 2016 [0021]
- Real-Time Anomaly Detection for Streaming Analytics. **AHMAD, S. ; PURDY, S.** arXiv:1607.02480. Cornell University Library, July 2016 [0021]
- Evaluating Real-time Anomaly Detection Algorithms - the Numenta Anomaly Benchmark. **LAVIN, A. ; AHMAD, S.** arXiv:1510.03336. Cornell University Library, October 2015 [0021]
- The HTM Spatial Pooler: A Neocortical Algorithm for Online Sparse Distributed Coding. **CUI, Y. ; AHMAD, S. ; HAWKINS, J.** bioRxiv. Cold Spring Harbor Laboratory, February 2017 [0021]
- **CUI, Y. ; AHMAD, S. ; HAWKINS, K.** Continuous Online Sequence Learning with an Unsupervised Neural Network Model. *Neural Computation*, November 2016, vol. 28 (11) [0021]
- **HAWKINS, J. ; AHMAD, S.** Why Neurons Have Thousands of Synapses, A Theory of Sequence Memory in Neocortex. *Frontiers in Neural Circuits*, March 2016, vol. 10, 1-13 [0021]
- *Sparse Distributed Representations*, 29 March 2017 [0025]
- **OLSHAUSEN, B.A. ; FIELD, D.J.** Sparse coding with an overcomplete basis set: A strategy employed by V1?. *Vision Research*, 1997, vol. 37, 3311-3325 [0025]